## Claims

What is claimed is:

1) A system for generating specialized executables, comprising:

a virtual subsystem that processes a generic code image; and

a log to store information relating to an operating environment of the virtual subsystem, the logged information is employed as feedback to generate a native executable, the native executable is utilized to provide improved performance of the virtual subsystem.

2) The system of claim 1, wherein the native executable is selected for execution by the virtual subsystem by matching a current environment setting with the logged information.

3) The system of claim 1, further comprising an image repository to store 1 through N specialized native images, wherein N is an integer.

4) The system of claim 3, wherein the image repository is a local or remote database.

5) The system of claim 1, further comprising at least one of a local or remote data log to store the logged information.

6) The system of claim 5, wherein a data log stores 1 though N environment parameter descriptions associated with 1 to N encountered images, wherein N is an integer.

7) The system of claim 1, wherein the virtual subsystem includes at least one of an execution engine and a virtual machine, and wherein the generic code image is an intermediate language image.

8) The system of claim 7, wherein the virtual subsystem employs Just-In-Time compilation on the generic code image.

9) The system of claim 7, wherein the virtual subsystem at least one of loads related executables, organizes data fields and methods within the generic image, analyzes intermediate language formats and generates native platform code.

10) The system of claim 7, wherein the virtual subsystem creates a generic native image at about the time the generic code image is installed.

11) The system of claim 1, wherein the logged information includes a set of information to enable executable images according to at least one of a particular user, a method of invocation and according to a usage pattern.

12) The system of claim 1, wherein a native code image is generated by the virtual subsystem if a native executable is unavailable.

13) The system of claim 1, further comprising an image generator for processing the logged information feedback and generating the native executable.

14) The system of claim 13, wherein the image generator further comprises at least one of a compiler and a code processor.

15) The system of claim 14, further comprising an image processing tool to read the logged information and associate one or more environmental settings with one or more related images encountered during virtual subsystem execution.

16) The system of claim 1, wherein the logged information includes parameters relating to at least one of an operating system version, a processor type, virtual system version, processor specifiers, developer parameters, domain flags, security information, binding information, administrative flags, and profile information associated with the operating environment of the virtual subsystem.

17) The system of claim 16, wherein the parameters are associated with an identifier to enable selection the native executable.

18) The system of claim 16, wherein the developer parameters describe at least one of debug options, compiler switch settings and information relating to preferences of a user.

19) A computer-readable medium having computer-executable components for executing the system of claim 1.

20) A method to generate runtime code, comprising:
determining a first code image associated with a possible runtime environment;
executing the first code image in the runtime environment; and
generating runtime feedback associated with the first code image to adjust a subsequent code image according to the runtime environment.

21) The method of claim 20, further comprising,
generating a specialized executable from the subsequent code image.

22) The method of claim 21, further comprising,
storing the specialized executable in an image repository.

23) The method of claim 21, further comprising,
processing a generic image utilizing standard compilation techniques.

24) The method of claim 23, further comprising,

logging operating environment information during processing of the generic image.

25) The method of claim 23, further comprising,

building the specialized executable from the environment information.

26) The method of claim 25, further comprising,

selecting the specialized executable by matching a current environment setting with the logged environment information.

27) The method of claim 20, further comprising at least one of:

loading related executables;

organizing data fields and methods within the first code image; and

analyzing intermediate language formats to generate native platform code.

28) A system to generate runtime code, comprising:

means for determining a specialized code image associated with a possible runtime environment;

means for executing the specialized code image in the runtime environment; and

means for generating runtime feedback to adjust a subsequent specialized code image according to the runtime environment.

29) The system of claim 28, further comprising,

means for generating a specialized executable from the subsequent code image.

30) A signal facilitating generation of specialized executables, comprising:

a signal for communicating between one or more components of a virtual system, the virtual system processing a generic code image and logging information relating to an operating environment of the virtual system *via* the signal;

wherein the logged information is employed as feedback across the signal to generate a specialized native executable, the specialized native executable is utilized to provide improved performance of the virtual system.

31) The signal of claim 31, wherein the signal is communicated over at least one of a network system and a wireless system.

32) A computer-readable means having stored thereon a data structure, comprising:

a first data field having parameters relating to at least one of an operating system version, a processor type, a virtual system version, and a processor specifier;

a second data field having at least one of a developer parameter, a domain flag, a security information field, and a binding information field; and

a third data field having at least one of an administrative flag and a profile information field associated with the operating environment of a virtual system.

33) A virtual software system, comprising:

an execution engine that processes an Intermediate Language (IL) image, the execution engine generating operating environment data while processing the IL image; and

a specialized executable image generated at least in part from the operating environment data, the specialized executable image stored in a repository of one or more other specialized executable images;

wherein the execution engine selects at least one specialized executable image from the repository if the at least one specialized image matches present operating environment data.